1) the proxy facilitates a move/copy/trash operations on an open and active document; and

2) the proxy can also be dropped onto devices and appliances, without closing the document or removing the proxy from the document.

Stationery Pad Object.

If a stationary pad object is opened into the workspace, then the icon creates a copy of the stationery pad base document and opens the document. If the stationery pad object is opened into a window with a proxy, then the stationery pad object cannot be in an opened state in a normal document sense, because opening it causes a copy (a normal document and not a stationery pad) to be created and opened within the document containing the proxy. Another variation is when a Show Properties window for a stationery pad has a proxy, it can be opened (to copy off and open a stationery document instance) from the Show Properties window.

Printer object.

A printer object is opened in the workspace, then it becomes an icon to which you can drag documents to have an associated hardware printer device print a document. If the printer object is opened to a window with a proxy, the printer object may opens into a window that shows the state of queued jobs, the state of the printer (like paper being out), or a control panel for special printer functions. Including a proxy in the window allows the user to continue to use the same interface for dragging a job to the printer, namely by dragging a document icon to the proxy icon.

FIG. **10** is a flowchart setting forth the detailed logic associated with proxy operations in accordance with a preferred embodiment. Processing commences at function block **1000** when a mouse down occurs. Then a test is performed at decision block **1002** to determine if the cursor is located in an event area when the mouse down event was detected. If so, then the event is posted and handled as shown in function block **1004** and another test is performed at decision block **1010** to determine if a draggable entity has been selected. If not, then control is passed to another interaction as shown in function block **1012**, and processing is returned to function block **1000** to await the next mouse down event. If a draggable entity is detected at decision block **1010**, then a drag and drop interactor is created at function block **1020** and the next event associated with the next mouse action is received at function block **1030**, and a test is performed at decision block **1040** to determine if a new acceptor has been positioned under the icon. If so, then post drag exit event as shown in function block **1042** and pass control to decision block **1050** to determine if the same acceptor is positioned under the selected icon. If it is not the same acceptor, then a drag enter event is posted at function block **1070** and control is passed to decision block **1060** to determine if a mouse drop event has occurred. If so, then perform the acceptor function as shown in function block occurred. If so, then perform the acceptor function as shown in function block **1080**. If not, then pass control to **1030** to process the next event.

A background grammar analysis tool is implemented as a software icon to which you drag text documents to invoke an analysis tool to analyze a document that is dropped on the icon representative of the analysis tool. If the opened appliance window has a proxy, then the proxy provides the interface for accepting more documents to analyze while the current results are displayed in the same window. Another example could be a spell check operation invoked similarly.

While the invention has been described in terms of a preferred embodiment in a specific system environment, those skilled in the art recognize that the invention can be practiced, with modification, in other and different hardware and software environments within the spirit and scope of the appended claims.

Having thus described our invention, what we claim as new, and desire to secure by Letters Patent is:

1. A framework for creating, displaying and managing a proxy associated with a document on a computer system having a memory, a display, a pointing device, an event handler for detecting hit events, mouse down events, and mouse up events and for generating coordinate data associated with each event, and a view manager including a means for registering a target view, indicating the coordinates where on the display the target view is displayed, the framework comprising:

(a) class information stored in the memory defining a proxy object, which includes
  first program code means for associating the proxy object with a first document;
  graphic data indicative of the proxy object;

(b) class information stored in the memory defining a draggable proxy object, which includes
  second program code means for associating the draggable proxy object with the proxy object;
  a list of candidate types supported by the first document;
  third program code means, responsive to a mouse up event and cooperating with the view manager, for determining whether the coordinate data of the mouse up event is over a registered target view and for issuing a drop message having the list of candidate types; and graphic data indicative of the draggable proxy object;

(c) class information stored in the memory defining a second document as a document container object from which a second document is created, the second document including
  a data structure for containing content data of the second document,
  fourth program code means for displaying a view of the content data on the display,
  fifth program code means for displaying a view of the proxy object, using the graphic data of the proxy object,
  sixth program code means, responsive to a hit event on the proxy view and a mouse down event, for creating a draggable proxy object from the class information defining a draggable proxy object and for displaying a view of the draggable proxy, using the graphic data of the draggable proxy object; and

(d) class information stored in the memory defining a target object, which includes
  seventh program code means for displaying a view of the target object and for registering the target view with the view manager; eighth program code means, responsive to the drop message from the third program code means, for choosing a preferred type from the list; ninth program code means, responsive to the drop message from the third program code means, for performing a predetermined operation on the document associated with the proxy object utilizing the chosen type.

2. The framework of claim 1 wherein the list of candidate types indicates how the first document is accessible through a conversion process.